

Retinopathy

by Manikanta Reddy

Submission date: 23-Dec-2020 04:42PM (UTC+0530)

Submission ID: 1480836323

File name: Mini_Project_Report_1.docx (2.15M)

Word count: 1522

Character count: 8130

ABSTRACT

Diabetic Retinopathy (DR) is one of the major causes of blindness. This disease is mainly observed in diabetic patients now a days we are observing a rapid growth of patients suffering from diabetes. DR is one of the primary causes of blindness. DR is diagnosed manually by ophthalmologist which is a more time taking process and hence in our project we aimed in automating the diagnosis of the disease into various classes using Convolution Neural Network model.

Our dataset contains high resolution Retinal fundus images which contains five different classes (0-No DR, 1-Mild, 2-Moderate, 3-Sever, 4-Proliferative DR) according to the disease severity.

In this project, a custom Convolutional Neural Network is built.

Input : Eye images

Layers : 1) 4 convolutional layers

2) 6 activation layers

3) 2 pooling layers

4) 3 dropout layers

5) 1 flatten

6) 2 dense layers These layers will extract the characteristics of the retinal Fundus photographs by using activation functions.

Activation

Functions : 1) Relu

2) Soft max

Optimizer : Adam Optimizer

Further model is trained for 10 epochs and validated against 30% of dataset that produced an accuracy of 72%.

KEY WORDS: : Diabetic Retinopathy, Convolutional Neural Networks, Retinal Fundus Images , relu , softmax , adam , sparse_categorical_crossentropy

Summary of the base paper

²Diabetic retinopathy (DR) is a vital reason of blindness worldwide. In the initial stages it is very hard to be detected and even for the experts, diagnostic procedure is time-taking. Therefore, a computerised treatment based on deep learning algorithms is suggested to automatically treat the referable diabetic retinopathy by categorizing color retinal fundus photographs into two grades.

Still now a days DR is screened manually by ophthalmologist which is a time taking process and hence this paper aims at automatic treatment of the disease into different stages using Convolution Neural Network model.

Our dataset is taken from Kaggle Diabetic Retinopathy provided by EyePACS and the size of our dataset is 500MB which contains high resolution of 600 fundus images of the retina which contains five different classes (0-No DR, 1-Mild, 2-Moderate, 3-Sever, 4-Proliferative DR) based on their severity. Each image in the dataset has a resolution of 4750X3168

In this project, a custom Convolutional Neural Network is built.

Input : Eye images

Layers : 1) 4 convolutional layers

2) 6 activation layers

3) 2 pooling layers

4) 3 dropout layers

5) 1 flatten

6) 2 dense layers These layers will extract the characteristics of the retinal Fundus photographs by using activation functions.

Activation

Functions : 1) Relu

2) Soft max

Optimizer : Adam Optimizer

After the model is built and compiled using desired parameters is finally trained for 10 epochs and trained against 70% of the dataset and validated against 30% of the dataset that produced an accuracy of 72%.

¹ **Merits and Demerits**

By, predicting Diabetic Retinopathy presence using CNN model is way better than diagnosing with clinical and medical procedures which is more time Consuming . So , we automated the diagnosis of Diabetic Retinopathy using CNN model which is very less time consuming and doesnot require any medical expertise due to this early prediction of Diabetic Retinopathy in the diabetic patients it helps them in curing their blindness problem at an early stage . It also helps diagnosing large number of patients in very short period of time with no need of consulting any doctor.

It does not give us 100% accuracy .So, there is a minute chance of wrong prediction of Diabetic Retinopathy in the patients which does not takes place in clinical and medical procedures.

.

CHAPTER 1

INTRODUCTION

Diabetic Retinopathy is one of the vital reasons of blindness and vision impairment worldwide. The reason of Diabetic Retinopathy is the increase of sugar levels in the blood of diabetic patient. In 2015, 0.4 million case of blindness and 2.6 million cases of severe vision impairment globally can be attributed to it. Diagnosis of Diabetic Retinopathy primarily depends on careful observation and analysis of retinal fundus photographs which is more time taking for experienced experts. Therefore , an automated treatment of DR is very important to detect DR in very short time which further helps in improving early stage DR identification their by reducing the number of blindness cases worldwide.

A Deep learning based method is suggested to automatically categorize the Retinal fundus images into five different classes from 0-4 i.e, from with-without RDR respectively.

- 0-No RDR
- 1-Mild RDR
- 2-Moderate RDR
- 3-Severe RDR
- 4-Proliferated RDR

CHAPTER-2 DATASET

The dataset used in this project is taken from Kaggle Diabetic Retinopathy provided by EyePACS and the size of our dataset is 500MB which contains high resolution of 600 fundus images of the retina which contains five different classes (0-No DR, 1-Mild, 2-Moderate, 3-Sever, 4-Proliferative DR) based on their severity . Each image in the dataset has a resolution of 4750X3168

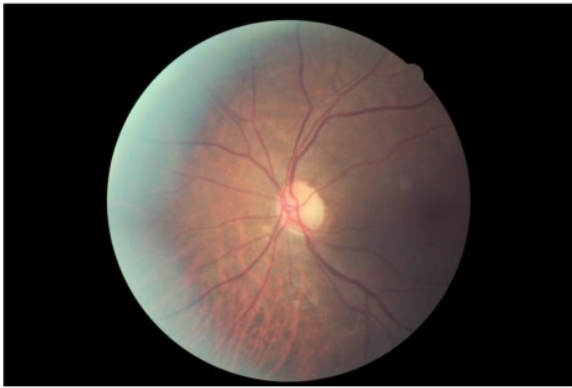


Fig 2.1. Eye image of No RDR



Fig 2.2. Eye image of Mild RDR

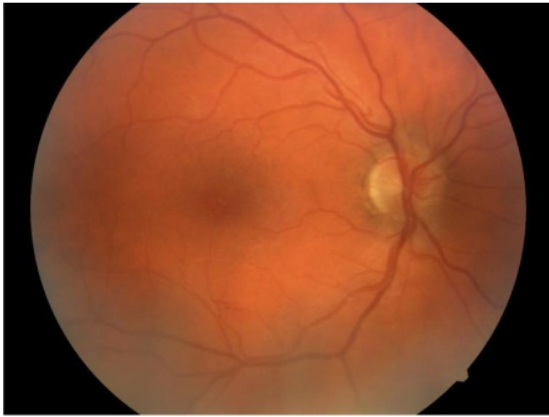


Fig 2.3.Eye image of Moderate RDR

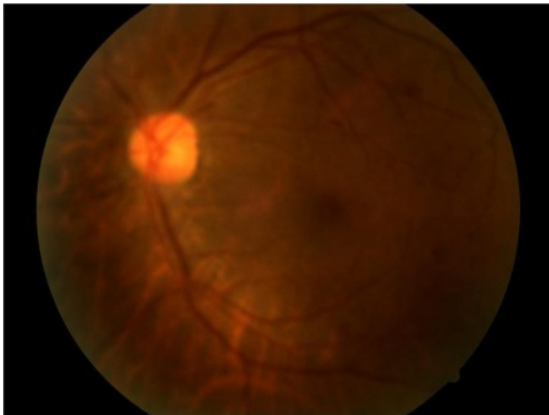


Fig 2.4.Eye image of Severe RDR



Fig 2.5 Eye image of Proliferated RD

CHAPTER-3 METHODOLOGY

Before Artificial Intelligence came into existence ,classification of medical images are done using edge detection filters and some mathematical function. But Now, medical images are classified by using Deep Neural Networks .A Neural Network can be built from scratch. This study uses a custom convolutional neural network architecture which consists of 18 convolution layers, that includes 4 convolution layers (2 Layers with 32 filters each of 3X3 size and 2 Layers with 64 filters each of 3X3 size) 6 Activation Layers (first five are 'relu' and the last Layer is 'softmax') 2 Pooling Layers (Max pooling with pool size of 2X2) 3 Dropout Layers (0.2 each) 1 Flatten Layer and 2 Dense Layers (with one 512 neurons and the other of 5 neurons) The proposed model should even identify minor changes in the images. The proposed model consists of 18 convolution layers .

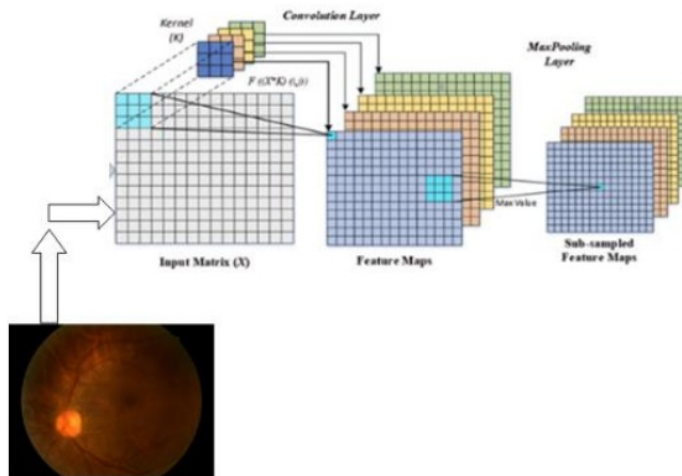


Fig 3.1.A Schematic Representation of convolution and Max pool layers

Number of Layer	Layer Type	Output shape	Parameters
1	Conv2d	[297,297,32]	896
2	Activation	[297,297,32]	0
3	Conv2d	[295,295,32]	9248
4	Activation	[295,295,32]	0
5	MaxPooling 2D	[147,147,32]	0
6	Dropout	[147,147,32]	0
7	Conv2d	[145,145,64]	18496
8	Activation	[145,145,64]	0
9	Dropout	[145,145,64]	0
10	Conv2d	[143,143,64]	36928
11	Activation	[143,143,64]	0
12	MaxPooling2D	[71,71,64]	0
13	Flatten	[322624]	0
14	Dropout	[322624]	0
15	Dense	[512]	165184000
16	Activation	[512]	0
17	Dense	[5]	2565
18	Activation	[5]	0

Table 3.1: The Layers and The Layers Parameters of the proposed model

Total Params 165:252:133

Trainable Params 165:252:133

Non-Trainable Params 0

CHAPTER-4

SOURCE CODE

```
File Edit Selection View Go Run Terminal Help
dummy.ipynb - Inception Train - Visual Studio Code
Trusted Jupyter Server: local Python 3: Idle

[1] In:
import PIL
import os
import os.path
from PIL import Image

f = r'C:\Users\Mani\Desktop\MiniProject\Diabetic_Retinopathy_Detection-master\data\data\input'
for file in os.listdir(f):
    f_img = f+"/"+file
    img = Image.open(f_img)
    img = img.resize((299,299))
    img.save(f_img)

[2] In:
arr=[]
import numpy as np
from numpy import asarray
for file in os.listdir(f):
    f_img = f+"/"+file
    img = Image.open(f_img)
    numpydata = asarray(img)
    arr.append(numpydata)
print(arr[0].shape)
print(len(arr))

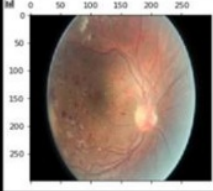
Python 3.8.0 64-bit @ 9.6.0
```

```
File Edit Selection View Go Run Terminal Help
dummy.ipynb - Inception Train - Visual Studio Code
Trusted Jupyter Server: local Python 3: Idle

300
X=arr

[3] In:
X=arr

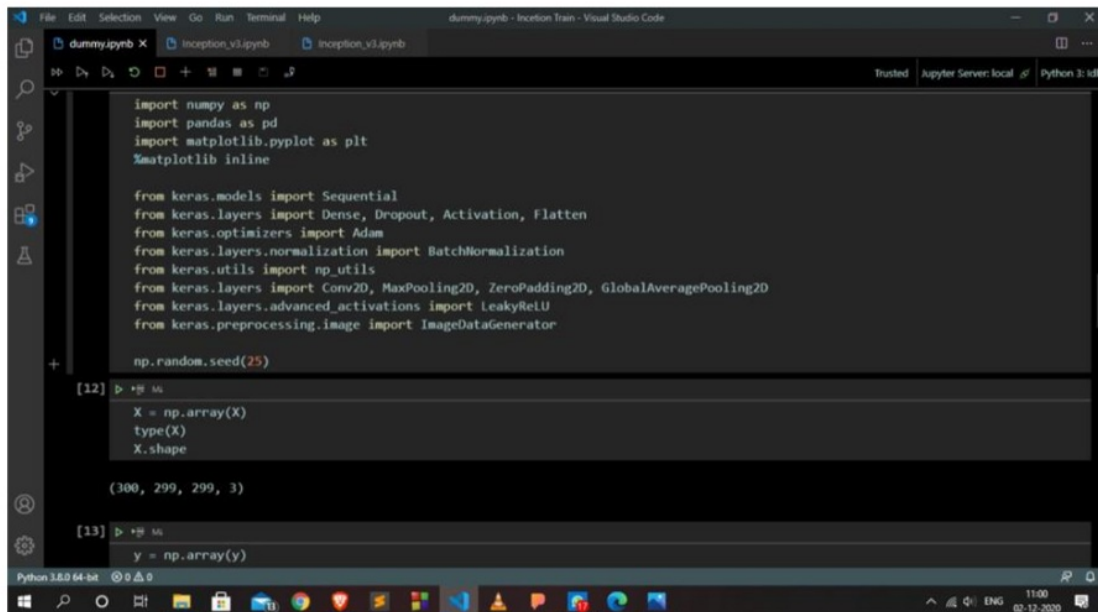
[4] In:
import matplotlib.pyplot as plt
plt.imshow(X[0])

<matplotlib.image.AxesImage at 0x2a0ffe2cc70>


[5] In:
y=[]
for i in range(9):
```

Pre-processing implementation

Actually our original input images are in the form of RGB(4750X3168X3) and after performing the pre-processing on this input images our final output images that are suitable for our CNN model are in the form of RGB(299X299X3)



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.optimizers import Adam
from keras.layers.normalization import BatchNormalization
from keras.utils import np_utils
from keras.layers import Conv2D, MaxPooling2D, ZeroPadding2D, GlobalAveragePooling2D
from keras.layers.advanced_activations import LeakyReLU
from keras.preprocessing.image import ImageDataGenerator

np.random.seed(25)
```

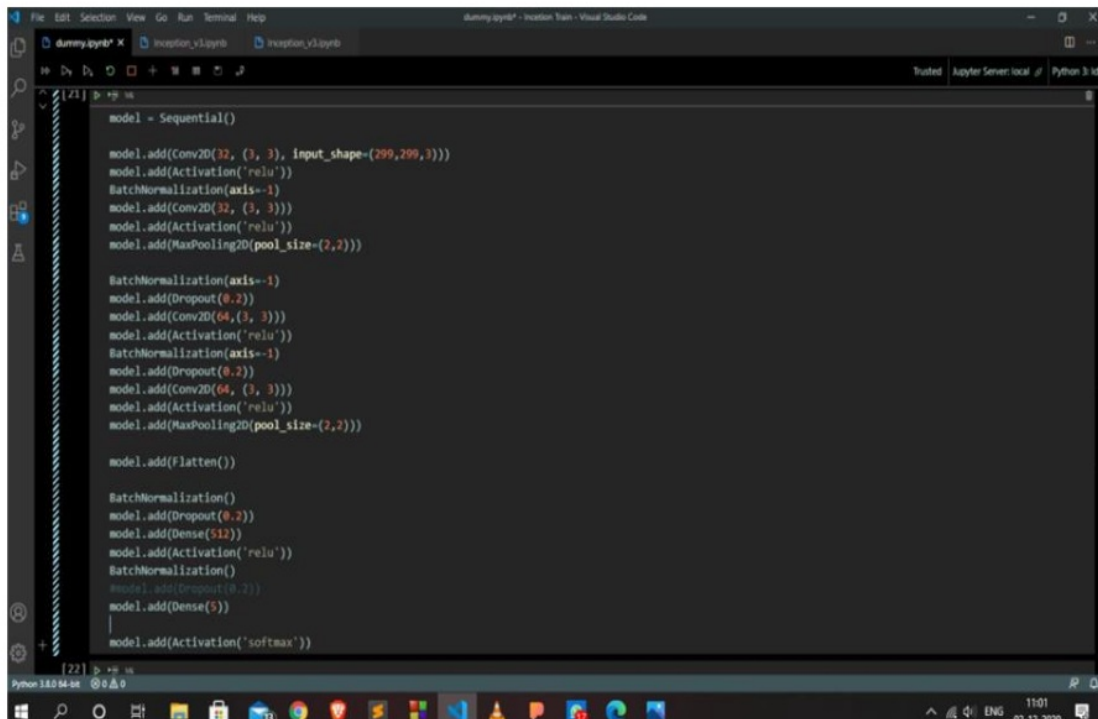
[12] In [12]:

```
X = np.array(X)
type(X)
X.shape
```

(300, 299, 299, 3)

[13] In [13]:

```
y = np.array(y)
```



```
model = Sequential()

model.add(Conv2D(32, (3, 3), input_shape=(299, 299, 3)))
model.add(Activation('relu'))
BatchNormalization(axis=-1)
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

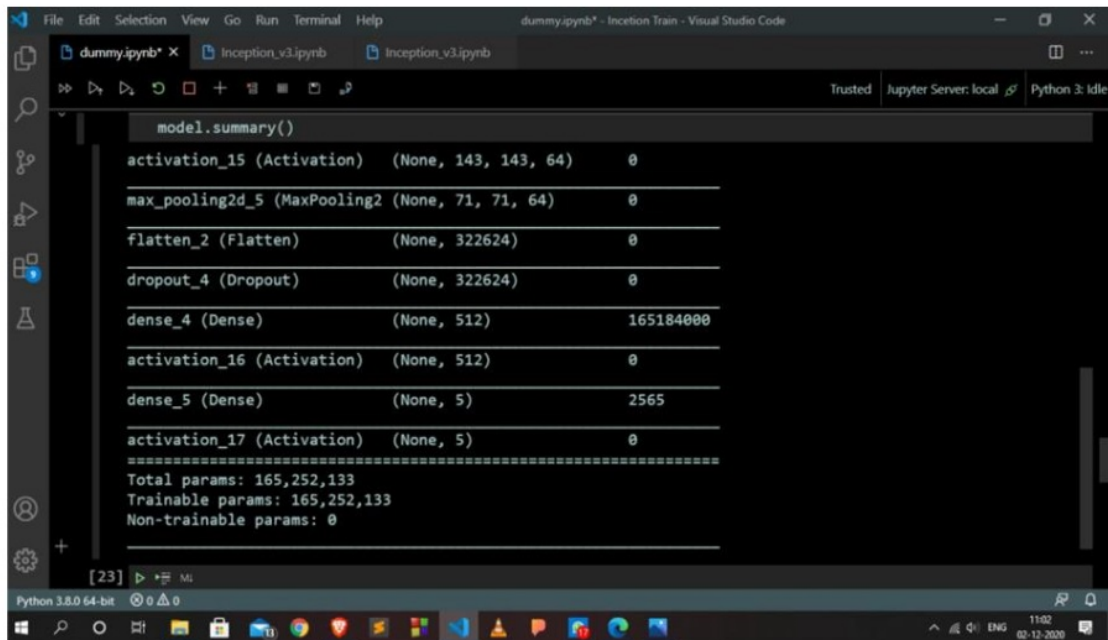
BatchNormalization(axis=-1)
model.add(Dropout(0.2))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
BatchNormalization(axis=-1)
model.add(Dropout(0.2))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

BatchNormalization()
model.add(Dropout(0.2))
model.add(Dense(512))
model.add(Activation('relu'))
BatchNormalization()
model.add(Dropout(0.2))
model.add(Dense(5))
model.add(Activation('softmax'))
```

[22] In [22]:

Model Building Implementation

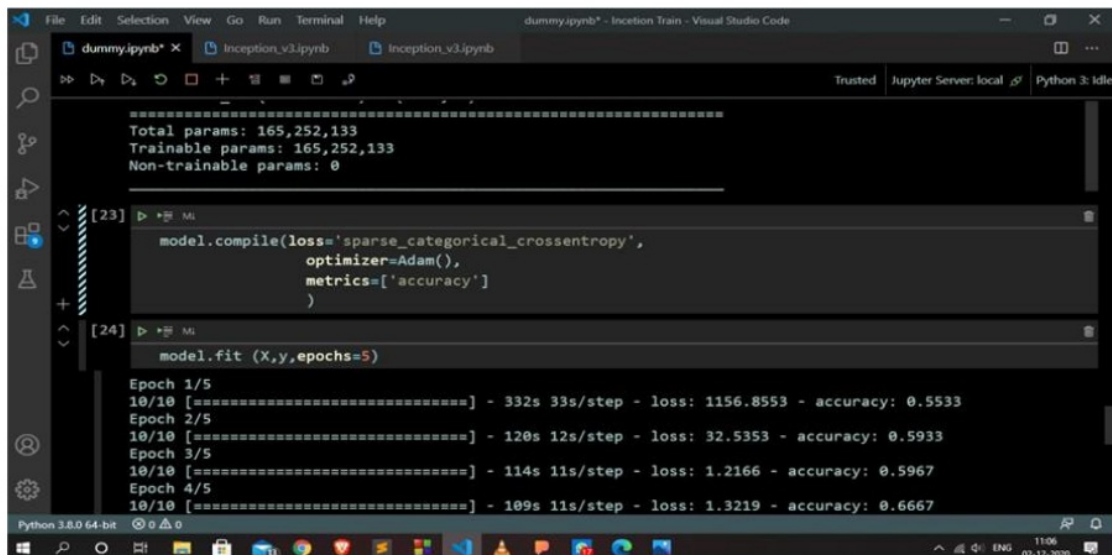


```
model.summary()

activation_15 (Activation) (None, 143, 143, 64) 0
max_pooling2d_5 (MaxPooling2 (None, 71, 71, 64) 0
flatten_2 (Flatten) (None, 322624) 0
dropout_4 (Dropout) (None, 322624) 0
dense_4 (Dense) (None, 512) 165184000
activation_16 (Activation) (None, 512) 0
dense_5 (Dense) (None, 5) 2565
activation_17 (Activation) (None, 5) 0
=====
Total params: 165,252,133
Trainable params: 165,252,133
Non-trainable params: 0
```

Model Summary

Model Compilation



```
=====
Total params: 165,252,133
Trainable params: 165,252,133
Non-trainable params: 0

[23] > * ML
model.compile(loss='sparse_categorical_crossentropy',
              optimizer=Adam(),
              metrics=['accuracy'])

[24] > * ML
model.fit(X,y,epochs=5)

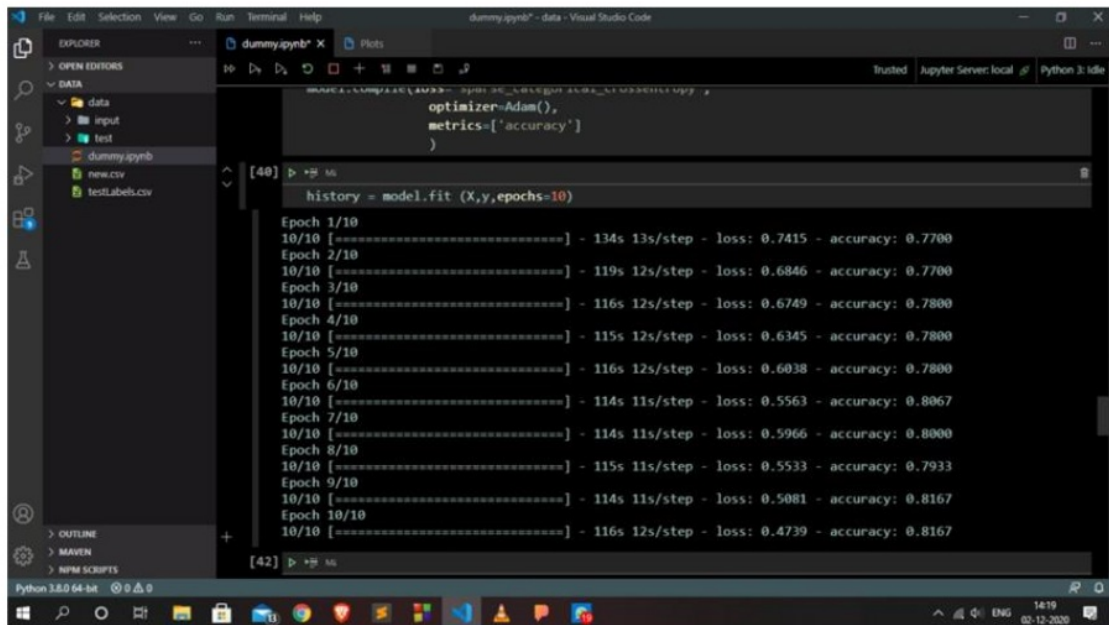
Epoch 1/5
10/10 [=====] - 332s 33s/step - loss: 1156.8553 - accuracy: 0.5533
Epoch 2/5
10/10 [=====] - 120s 12s/step - loss: 32.5353 - accuracy: 0.5933
Epoch 3/5
10/10 [=====] - 114s 11s/step - loss: 1.2166 - accuracy: 0.5967
Epoch 4/5
10/10 [=====] - 109s 11s/step - loss: 1.3219 - accuracy: 0.6667
```

Loss Function: Sparse Categorical Cross Entropy

Optimizer Function : Adam()

Metrics : Accuracy

Model Training



The screenshot shows a Jupyter Notebook with the following code and output:

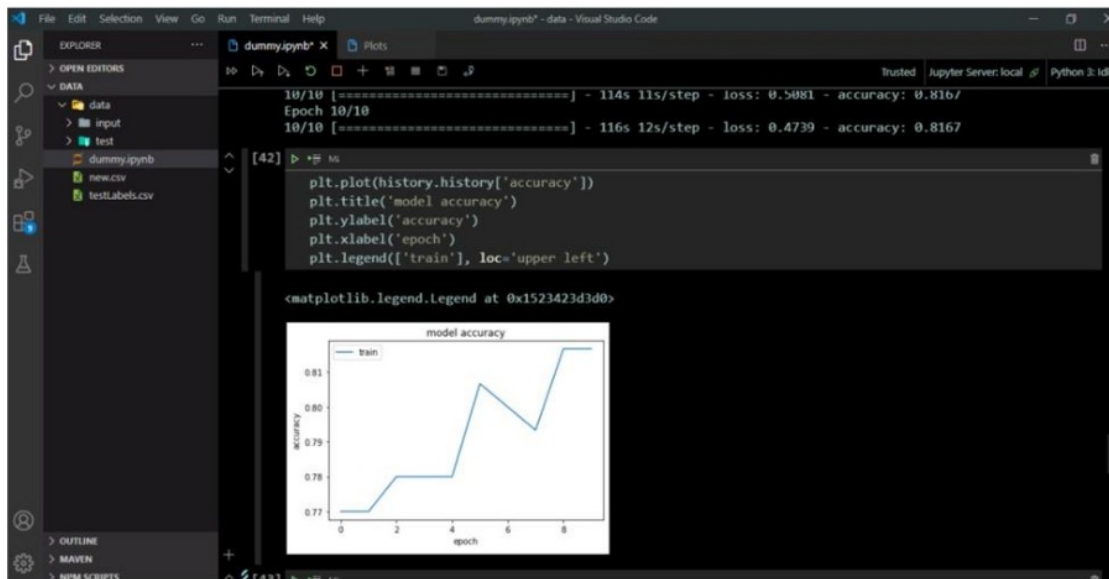
```
optimizer=Adam(),  
metrics=['accuracy']  
)  
  
[40]: history = model.fit(X,y,epochs=10)
```

The output displays the training progress for 10 epochs:

Epoch	Time	Step	Loss	Accuracy
1/10	134s	13s/step	0.7415	0.7700
2/10	119s	12s/step	0.6846	0.7700
3/10	116s	12s/step	0.6749	0.7800
4/10	115s	12s/step	0.6345	0.7800
5/10	116s	12s/step	0.6038	0.7800
6/10	114s	11s/step	0.5563	0.8067
7/10	114s	11s/step	0.5966	0.8000
8/10	115s	11s/step	0.5533	0.7933
9/10	114s	11s/step	0.5081	0.8167
10/10	116s	12s/step	0.4739	0.8167

Epochs: Number of times all the input images are trained

Model Evaluation



```
[43] > model.evaluate(X_test,y_test)

10/10 [=====] - 20s 2s/step - loss: 1.3392 - accuracy: 0.7200
[1.3391727209091187, 0.7200000286102295]
```

Accuracy:72.0

1 CHAPTER-5

RESULTS

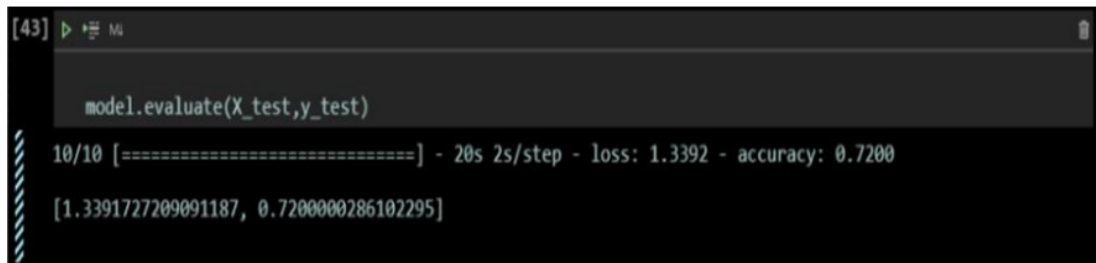
There are two types of classification for predicting Diabetic Retinopathy .They are:

1. Binary Classification

2. Multi class classification (0-no , 1-Mild , 2-Moderate , 3-Severe , 4-Proliferated)

Multi class classification

- In our Dataset we have choosen 70% of the data as the training dataset and the remaining 30% of the data as the testing dataset
- The Accuracy we have obtained after training the model and evaluated against the validation data is 72.0%



```
[43] ▶ M4
model.evaluate(X_test,y_test)
10/10 [=====] - 20s 2s/step - loss: 1.3392 - accuracy: 0.7200
[1.3391727209091187, 0.7200000286102295]
```

Fig 5.1.Final Accuracy of the model obtained

We trained our CNN model for different loss functions and optimizer algorithms but after the results we observed that when loss function= sparse_categorical_crossentropy and optimizer algorithm=Adam() is the best fit for our CNN model built

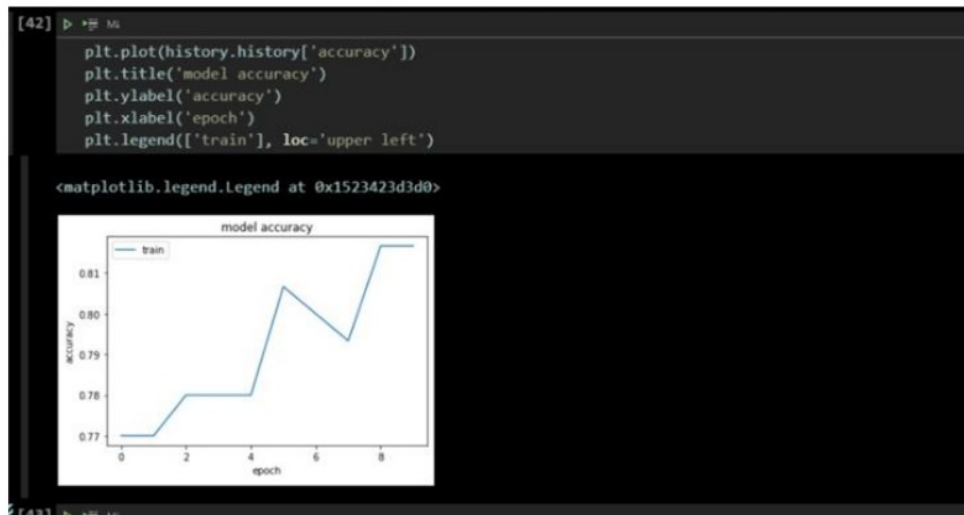


Fig.5.2.Accuracy curve plotted against epoch over time during training of the model

While training our model after plotting the curve against epoch and accuracy it is observed that for every epoch the accuracy score is steadily increasing till the epoch-10. After this there is no fluctuation in the accuracy score and hence, it is inferred that training model for 10 epochs is enough.

CHAPTER-6

CONCLUSION AND FUTURE WORK

²In this Project , a novel Convolutional Neural Network Model to Automatically Diagnosis ²the Retinal Diabetic Retinopathy based on DL is developed. Our model takes Retinal fundus image ²as inputs and predicts the possibility of RDR. In our dataset we have choosen ⁴70% of the data as the training dataset and the rest 30% of the data as the testing dataset. We have trained our novel CNN model with ten Epochs and a graph is plotted against number of epochs and accuracy for obtaining the correct epoch number. Our CNN model achieves an accuracy after training the model and evaluated against the validation data is 72.0%

Generally a Deep Convolutional Neural Network requires a large dataset to get better results, but the dataset we have choosen is not large enough to obtain good results. So, a pretrained model on relatively smaller dataset comparitavely produces a better results. Hence, several pretrained models such as VGG-16, ResNet-50, Inception V3 and EfficientNet will produce better results even on a smaller dataset. So, to make use of these already pretrained models transfer learning method is used.

Retinopathy

ORIGINALITY REPORT

16%

SIMILARITY INDEX

2%

INTERNET SOURCES

6%

PUBLICATIONS

13%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to SASTRA University

Student Paper

6%

2

Xianglong Zeng, Haiquan Chen, Yuan Luo, Wenbin Ye. "Automated Diabetic Retinopathy Detection Based on Binocular Siamese-Like Convolutional Neural Network", IEEE Access, 2019

Publication

4%

3

Submitted to University of Sydney

Student Paper

2%

4

Submitted to University of Sheffield

Student Paper

2%

5

Submitted to Monash University

Student Paper

1%

6

Eva-Maj Malmström, Mikael Karlberg, Agneta Melander, Måns Magnusson, Ulrich Moritz. "Cervicogenic dizziness – musculoskeletal findings before and after treatment and long-term outcome", Disability and Rehabilitation,

1%

2009

Publication

Exclude quotes On

Exclude bibliography On

Exclude matches < 3 words

Retinopathy

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16